

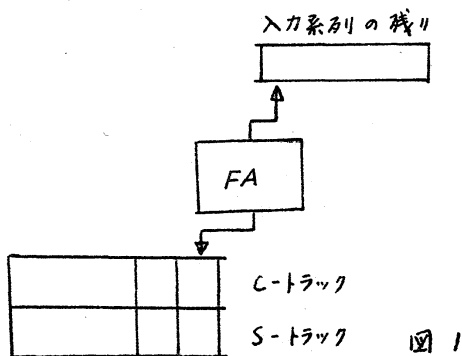
ANALYZERの簡単化

阪大 基礎工 菊野 亨

1. まえがき

コンパイラにおけるいわゆる“解析部”(analysis part)の主な目的は、与えられたソースプログラムを分析し、パラメータをもってセマンティックルーチンをつぎつぎと呼ぶことである。たとえば、記号表へのいくつかのポインタをパラメータとして内部コード発生ルーチンをつぎつぎと呼ぶ。

本稿では、このようなアナライザのモデルとして、一定量おじの先よめを許した決定性のプッシュダウンオートマトンを考える。(図1) プッシュダウンスタックは、動作を決



定するために用いるコントロール記号を書きこむC-トラックと、呼ぶべきセマンティックルーチンに対するパラメータを書きこむS-トラックに

わかれてゐる。このアナライザのコントロール部はフローチャートの形で表現される。

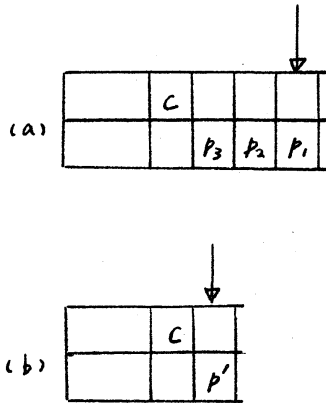


図-2

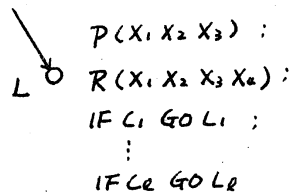


図-3

今かりにプッシュダウンスタックが図2(a)のようであるとせ、コントロール

がフローチャート上で図3の節点Lにきた場合を考える。Pは新しいパラメータを求めるルーチン名である。まず

現在のS-トラックのtopの3個の P_1, P_2, P_3 をそれぞれ X_1, X_2, X_3 に代入するパラメータとしてルーチンPを呼ぶ。

新しいパラメータ P' を求める。次に X_1, X_2, X_3, X_4 にそれぞれパラメータ P_1, P_2, P_3, P' を代入しセマンティックルーチンRを呼ぶ。スタックは長さ2だけ短くなる。

P' をS-トラックのtopに書きこみ(図2(b))、左となりのコマのコントロール記号Cが C_i に等しければ、次は節点 L_i から実行する。フローチャートでは、このようにルーチンを呼ぶことを指定した節点のほか、C-トラックのtopにコントロール記号を書きこむ動作を指定した節点や、先よみのヘッドを一つ右へ動かし、入力記号の先よみを指定する節点などもある。

二つのアナライザA, Bは、任意の入力系列(ソースフロ

グラムまたは語い解析を終えたもの) λ に対し、 A が λ を拒否するならば B も拒否し、 A が受理するならば B も受理し、かつ A と B が同じパラメータで同じセマンティックルーチンをつぎつぎと呼んだとき、等価であるという。C-トラックにコントロール記号を書きこむ動作や、先読みをする動作などは、等価性には直接関係はない。等価性に意味があるのは、パラメータ $p_1, \dots, p_k$ をもってセマンティックルーチン R を呼ぶという動作のみである。すなわち、セマンティックルーチン R をパラメータ $p_1, \dots, p_k$ をもって呼ぶ動作を、 $CALL R(p_1, \dots, p_k)$ と書くことにすると、アナライザの全動作系列のうちでセマンティックルーチン R を呼ぶ動作だけを抜き出した系列 $\dots, CALL R(p_1, \dots, p_k) \dots$ が (記号の系列として) 同じであるとき等価であるという。パラメータ $p_1, \dots, p_k$ をもってルーチン P を呼んで新しいパラメータ p' を求めるとき、 p' はルーチン名 P とそのときのパラメータの値 $p_1, \dots, p_k$ とその時点までに読み込んだ入力系列によって一意にきまるものとし、それらが全部同じときのみ同じ p' が求まるとする (いわゆる自由解釈)。

本稿では、まず、与えられた二つのアナライザが等価であるかどうか決定可能であることを述べ、次に、与えられたアナライザと等価なアナライザのうちで、もっとも簡単なもの (直観的に言え、プログラムの書きとしてもっとも速く実

行するプログラムの中で、もっとも短かいプログラム、詳しくは後述)を求める方法を述べる。

なお、特定の文法 G を対象にし、与えられる入力系列が G で導出されるセンテンスかどうかを判別し、もしセンテンスならその導出不(節点のラベルは G の非終端記号または G のルール名)のすべてを与えるものを " G に対するパーサ" という。パーサの簡単化については文献(2)-(5)などで扱われているが、本稿でいうアナライザの簡単化はパーサのそれとは異なる。

2. アナライザのモデル

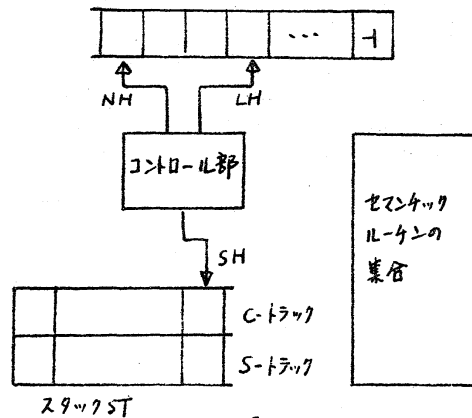


図-4

便宜的に図4のようなモデルを考えると、動作開始時には、ヘッド NH , LH と入力テープの左端に、ヘッド SH はスタックの底にある。動作中の NH と LH の距離は、高々 K

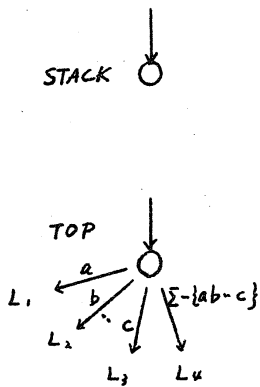
コマとする。アナライザのコントロールは計算機のプログラムに対応した次のようなフローチャートで書かれる。フローチャートの各節点には、そこにコントロールがきたときにとるべき基本動作が書かれている。

入力アルファベットの集合 Σ の部分集合 M に対し、 $\Sigma - M$ を一つの記号とみて、この記号の集合を $\mathcal{M}$ とする。

$$\mathcal{X} = \{\epsilon\} \cup X \cup X^2 \cup \dots \cup X^k \quad X \in \Sigma \cup \mathcal{M}$$

とする。 $X \in \Sigma \cup \mathcal{M}$ に対し、関数 λ を $X \in \Sigma$ なら $\lambda(X) = X$, $X = \Sigma - M \in \mathcal{M}$ なら $\lambda(X) = \{x \mid x \in \Sigma, x \notin M\}$ とする。

基本動作



(1) STACK SH を 1 コマ右へ動かして、 λ の C-トラックと S-トラックに NH のコマの入力記号をスタックする。† NH は 1 コマ右へ動く。

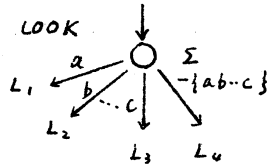
(2) TOP SH のコマの C-トラックの記号をみて、それにより、2 次の動作を定める。たとえば、図 4 の場合、その記号が a なら L_1 へ、b なら L_2 へ、...、c なら L_3 へ、それ以外なら L_4 へとコントロールが移る。(TOP はやはりとして STACK の直後だけとする)。

(3) WRITE(C) SH のコマの C-トラックにコントロール記号 c を書きこむ。

† 入力系列が語の解析を終えたものと考える場合は、たとえば、*identifier* であることを示す記号 I が C-トラックに、記号表へのポインタが S-トラックにスタックされる。



(4) AHEAD LH と 1 コマ 右へ動かす。



(5) LOOK LH の コマ の記号と MZ. (2) と同

じょうに次の動作を定める。

(6) RED(n); CALL(P,R); GO L 現在の S-トラ

ックの top の n 個の記号 ($p_1 \dots p_n$) をパラメ

ータとしてルーチン P を呼ぶ。新しいパラ

メータ p' を求める。パラメータ ($p_1 \dots p_n, p'$) を

使ってセマンティックルーチン R を呼ぶ。ス

タックは長さ (n-1) だけ短くなる。 p' と

S-トラックの top に書きこむ。次は節点 L から

実行する。

RED(n):
CALL(P,R);
GO L
(L は節点名, n は
ルーチン P のパラメータ数)

RED(n):
CALL(P,R);
IFC GO L₁;
:
IFC GO L₄;



(7) RED(n); CALL(P,R); IFC GO L₁; ...; IFC GO L₄

(6) と次のことを除いて同じ動作をする。

すなわち、スタックが長さ (n-1) だけ短か

くなら、後の左と右のコマの C-トラック

内のコントロール記号 C が C_i なら節点 L_i

から実行する。

(8) ACCEPT 入力系列と受理して停止する。

このときには、NH, LH はいずれも入力テープの右端の記号 + のコマにあり、SH は ST の底から 2 番目と指していることが保障されているものとする。

(9) REJECT 入力系列を拒否して停止する。

3. アナライザの等価性

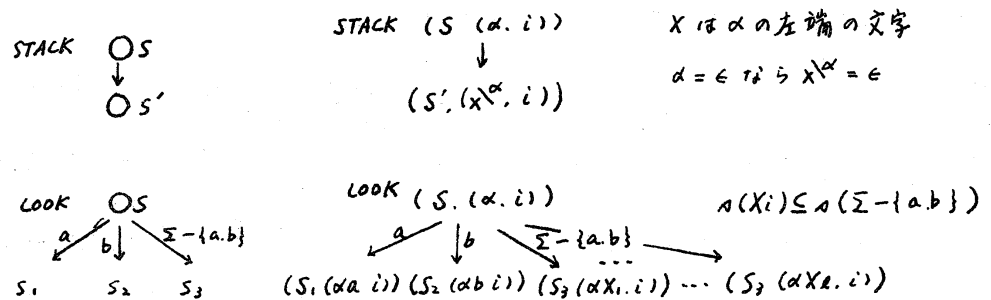
アナライザは、そのコントロール部のフローチャートで与えられる。そのフローチャートから一定の方法で文法 $G = (V_N, V_T, R, S)$ と写像 $h: V_N \rightarrow \tilde{V}_N$ を作る。 G の任意の導出木 t に対し (端点でない節点の) 各ラベル $X \in V_N$ を $h(X) \in \tilde{V}_N$ で置きかえて得られる木を簡単に $h(t)$ と書く。 $J(G, h) = \{ h(t) \mid t \in D(G) \}$ (ただし $D(G)$ は G の導出木の全体) とする。任意のアナライザ A, B に対しその方法で作った文法 G_A, G_B と写像 h について、 $J(G_A, h) = J(G_B, h)$ のとき、かつそのとき A と B は等価であるがなりたつ。

3.1 フローチャートの変換

アナライザ A はフローチャートの形で与えられる。これを $FL(A)$ とする。この $FL(A)$ から、各節点を分解して、先よみ情報 (先よみ系列と LH の位置) が一意に対応するようにする。これを $FL(A)'$ とする。つまり、 $FL(A)'$ の各節点は $(S, (\alpha, i))$ の形をしており、 $|\alpha| \leq k$, $\alpha \neq \epsilon$ のときは $i = |\alpha|$ か $|\alpha| + 1$, $\alpha = \epsilon$ のときは $i = 1$ とする。次にその手続を述べる。

$FL(A)$ の各節点 S について、 $\alpha \in \Sigma$, i のすべての組み合わせで節点 $(S, (\alpha, i))$ を作る。ラベルは S に書かれているもの

をそのうち書く。ただし $ACCEPT$ に対応する節点は $(S_F(t, 1))$ だけとする。(S_F は $FL(A)$ の $ACCEPT$ をラベルとして持つ節点)。次に、枝は節点のラベルに対応して、次のように結ぶ。



$TOP, WRITE(c)$ は $FL(A)$ の S から $S' \rightarrow$ 枝があれば、 $(S(\alpha, i))$ から $(S'(\alpha, i))$ へ、 $AHEAD$ の場合は $(S(\alpha, i))$ から $(S'(\alpha, i+1))$ へ枝を結ぶ。開始点は $(S_0(t, 1))$ (S_0 は $FL(A)$ の開始点) とし、最後に開始点から到達できない節点はすべて除く。

3.2 文法 G_A と写像 h

A の行なう基本動作の中で、 $STACK$ と $RED(n); CALL(P, R)$ の系列と $ACCEPT$ だけが等価性に意味がある。そこで $STACK$ や RED から次の $STACK$ や RED の間にある動作系列はまとめて、 Σ -系列とする。すなわち、 Σ -系列とは次のようなフローチャート $FL(A)'$ 上の節点の系列である。(1) 始まりは、開始点 $(S_0(t, 1))$ か、 RED の形が先か、 $STACK(TOP)$ の直後の節点で、(2) 終わりは、ラベルが $STACK$ か RED か $ACCEPT$ であるような節点で、(3) その途中には、二れらうちの基本動作をラベルとして持つ節

点を示すではない。 Σ -系列の始りの節点のラベルを $s(\Sigma)$, 終わりの節点のラベルを $e(\Sigma)$ とする。 Σ -系列の全体を Σ とする。

文法 $G_A = (V_N, V_T, R, S)$ を次のように $FL(A)'$ から Σ -系列を使, て作る。 $V_N = \{(\Sigma, X, \Sigma') \mid \Sigma, \Sigma' \in \Sigma, X \in \Sigma \cup N\}$, $N = \{(P, R) \mid P \text{ は ルーティン}, R \text{ は セマンティック ルーティン}\}$, $V_T = \Sigma$ とする。 R は次のように定義する。

各 $X \in V_T$ に対し, 以下の条件 (a) ~ (c) が成り立つように

$$(\Sigma, X, \Sigma') \rightarrow X$$

を R のルールとする。 (a) $s(\Sigma) = \text{STACK}(\text{TOP})$, (b) $FL(A)'$ で Σ から Σ' へ枝がある, (c) Σ の実行で スタック (判定された) 文字と Y とすると, $Y \in \Sigma$ ならば $X = Y$, $Y \in M$ ならば $X \in n(Y) \cap \Sigma$ 。

また, $(P, R) \in N$ に対し, 以下の条件 (d) ~ (g) が成り立つように

$$(\Sigma_{01} (P, R) \Sigma_{02}) \rightarrow (\Sigma_{11} X_1 \Sigma_{12}) (\Sigma_{21} X_2 \Sigma_{22}) \cdots (\Sigma_{n1} X_n \Sigma_{n2})$$

を R のルールとする。 (d) $s(\Sigma_{n2}) = \text{RED}(n); \text{CALL}(P, R);$ (e) $\Sigma_{01} = \Sigma_{11}$

$\Sigma_{12} = \Sigma_{21}, \dots, \Sigma_{n-1,2} = \Sigma_{n,1}$, $X_i \in \Sigma \cup N$ (f) $e(\Sigma_{21}) = e(\Sigma_{12}), e(\Sigma_{31}) = e(\Sigma_{22})$

$\dots e(\Sigma_{n,1}) = e(\Sigma_{n-1,2})$ if $\text{STACK}(\text{TOP})$ (g) $s(\Sigma_{n2}) = \text{RED}(n); \text{CALL}(P, R); \text{GOL}$

ならば Σ_{02} は節点 L から始まる Σ -系列, $s(\Sigma_{n2}) = \text{RED}(n); \text{CALL}(P, R); \text{IF}$

$C, \text{GOL}; \dots; \text{IF } C \text{ then } \text{GOL}$ ならば Σ_{11} はラベルが $\text{WRITE}(C)$ の節点を示

くんどおり, もし $C = C_j$ ならば Σ_{02} は節点 L_j から始まる Σ -系列。

$S = \{ (\xi, (PR), \xi') \mid \xi \text{ は } (S_0, (E, 1)) \text{ から始まる } \xi\text{-系列, } e(\xi') = \text{ACCEPT, } (PR) \text{ は } \xi' \text{ に入る直前に呼んでルータン} \}$

ここで、無駄な非終端記号、ルールは除く。次に準同型写像 $h: V_N \rightarrow V_T \cup N$ を任意の $(\xi, X, \xi') \in V_N$ に対し $X = a \in V_T$ なら $h(\xi, a, \xi') = a$, $X = (PR) \in N$ なら $h(\xi, (PR), \xi') = (PR)$ とする。

補題 1 アナライザ A と B が等価であるための必要十分条件は、上で述べた方法で作った文法 G_A, G_B , 写像 h について $J(G_A, h) = J(G_B, h)$ がなりたつことである。

$J(G_A, h) = J(G_B, h)$ がどうかは、文法の句構造的等価性の判定と類似の方法で決定できる。かえに、

定理 2 アナライザの等価性の問題は決定可能である。

4. アナライザの簡単化

ここでは、アナライザ A が与えられたとき、次の条件 $C1 \sim C5$ を満すようなアナライザ B を求める方法を述べる。

$C1$ A と B は等価である。

$C2$ 拒否はできるだけ早く行なう。すなわち、 A が受理する入力系列の全体を L とすると、入力系列 $x \notin L$ に対し

は、 B は次のような x の初期部分系列 y の最後の記号を見
 たり (先 $\downarrow$ まで、あるいは、先 $\downarrow$ までなければ $STACK$ 後) 時
 点で拒否する。 (1) y のおとに i かなる系列をつけ加えて
 L に属さない。 (2) y の任意の (真の) 初期部分系列 y' に
 対しては、 y' のおとに適當な系列をつけ加えれば L に属す
 る。

C3 先 $\downarrow$ まで知った情報は必ず利用する。すなわち、先
 $\downarrow$ までわかっている文字を再び見ることはないし、 TOP
 判定も行なわない。

C4 上の C1 ~ C3 を満たすアナライザの中で、先 $\downarrow$ までの長
 さが常に最小で、 $WRITE$ も必要最小限しか行なわない。

C5 上の C1 ~ C4 を満たすアナライザの中で、プログラムの
 の長さが最短。

4.1 最簡のアナライザの求め方 (あらまし)

S1 前節で述べた方法で、文法 G_A と写像 h とを求める。

S2 S1 で求めた文法 G_A と写像 h を変換して、次の条件 (a),

(b) を満たすような文法 $\bar{G}_A$ と写像 $\bar{h}$ を作る。

$$(a) J(G_A, h) = J(\bar{G}_A, \bar{h})$$

(b) $\bar{G}_A$ のルールで、もし $U \rightarrow \alpha, V \rightarrow \beta$ と右辺の同じ

ルールがあれば、必ず $\bar{h}(U) \neq \bar{h}(V)$ となる。

S3 文法 $\bar{G}_A$ に対するパーザを、前述の条件 C2, C3 を考慮

しながら Knuth と類似の方法で作る。

S4 S3 で求めたパーカと修正し、与えられたもののアナライカ A と等価なアナライカへ変換する。

S5 そのフローチャートと有限オートマトンとから、それとおおむね状態数最小のものとを見つける。これは Paull-Unger と類似の問題である。

4.2 文法の変換 (S2)

$\bar{G}_A$ と $\bar{h}$ は、 G_A と句構造的に等価な逆方向決定性文法 (backwards-deterministic grammar) の求め方において、 h で同じ値になる非終端記号だけを $\bar{h}$ と定めることによって得られる。

4.3 Knuth の K-状態

CFG を $G = (V_N, V_T, R, S)$ で表わす。一般性を失わずに次の仮定をおく。

- (i) ルール $\alpha \rightarrow \beta$ の右辺 β は ϵ でない。
- (ii) 各 $X \in V_N$ について $X \rightarrow \dots \rightarrow X$ でない。
- (iii) 無駄な非終端記号は全くない。

文法 G に対して $G' = (V_N \cup \{S_0\}, V_T \cup \{+\}, R \cup \{S_0 \rightarrow S+\}, S_0)$

を作る。 $S_0, +$ は $V = V_N \cup V_T$ に属さない新しい記号とする。

この G' を改めて $G = (V_N, V_T, R, S)$ とおく。 R の元は $S_0 \rightarrow S+$

と 0 とし $\#(R) - 1$ ($\#(R)$ は R の元の個数) までの数で適当に順序づけられているとする。 p 番目のルールを $X_p \rightarrow X_{p_1} \dots X_{p_{n_p}}$

で表わす. $\alpha \in V^+$, 整数 $k \geq 0$ に対し $H(\alpha) = \{t \in V_T^k \mid$

$G \text{ で } \alpha \xrightarrow{*} t\beta \text{ なる } \beta \in V^* \text{ が存在する.}\}$ とおく. Knuth (1)

にしたがって, K -状態 S の集合 $\mathcal{D}$ を次のように求める.

まず, $S_0 = \{[0, 0, 1]\}$, $\mathcal{D} = \{S_0\}$ とする. 任意の $S \in \mathcal{D}$,

$Y \in V$ について $S_Y = \{[p, j+1, \alpha] \mid [p, j, \alpha] \in S' \text{ かつ } X_{p,j+1} = Y\}$

を求め, 空でない場合は $\mathcal{D}$ へ入れる. ここで S' は $S' = S \cup$

$\{[0, 0, \beta] \mid X_{p,j+1} = X_0, \beta \in H(X_{p,j+2}, \dots, X_{p,n_p}, \alpha) \text{ なる } [p, j, \alpha] \in S' \text{ が存在する}\}$ を満たす最小の集合である.

K -状態 S , $0 \leq p \leq n_p$ に対し, $Z(S, p) = \{\alpha \mid [p, n_p, \alpha] \in S\}$ と

する. かつ $Z(S, \text{STACK}) = \{\beta \mid j < n_p, \beta \in H(X_{p,j+1}, \dots, X_{p,n_p}, \alpha) \text{ なる}$

$[p, j, \alpha] \in S' \text{ が存在する.}\}$ とする. $Z(S, \text{REJECT}) = V_T^{K-}$

$(Z(S, \text{STACK}) \cup Z(S, p_1) \cup \dots \cup Z(S, p_k))$ とする. G が $LR(K)$

文法なら, 各 S に対し $Z(S, p)$, $0 \leq p \leq \#(R) - 1$, $Z(S, \text{STACK})$,

$Z(S, \text{REJECT})$ は互いに素である. ここで, 各 $Z(S, X)$ (X は

$\text{RED}(p), \dots, \text{RED}(q), \text{STACK}$ あるいは REJECT) の各要素 α を次の

ような α' でおさかえる. (1) α' は α の初期部分系列, (2) α' は

他の $Z(S, X)$ の要素の初期部分系列ではない. これは, 必要

な先F4と受けとめるためである.

4.4 パーサの構成 (S3)

拒否とできるだけ早くおこなうため, 次の動作を選ぶとき

は, 拒否か STACK か $\text{RED}(p)$ かが一意に決まるだけの先F4と

するが、これは常に必要最小限の長さだけ行なうようなパーザと考える。しかも一度先読みで知った情報は必ず再利用する。 G_A に対するパーザの構成法については文献(7)参照(拒否と遅らせはいいことと、AHEADの動作と考慮する。)

4.5 アナライザへの変換 (S4)

G_A に対するパーザを $RED(p)$ と $RED(np):CALL(\bar{h}(X_p))$ でおきかえる。このコントローラ C_1 をもつアナライザを P_1 とする。 C_1 の $WRITE(C)$ や $RED(n):CALL(PR);IFC_1GOL_1;\dots;IFC_2GOL_2$ (または GOL) を単に $WRITE$ や $RED(n):CALL(PR)$ でおきかえ、枝のラベルを入力記号、その枝から次の分岐点または端点に至るまでの基本動作の系列と、その入力記号に対する出力記号と見て、これを有限オートマトン (F_1 とする) と考える。

4.6 アナライザの簡単化 (S5)

あるアナライザ P_2 のコントローラ C_2 が与えられたとする。 C_2 を前と同じように有限オートマトンとみなすものを F_2 とする。有限オートマトン F_1 のある状態 s から定義されているすべての入出力系列が F_2 のある状態 s' から定義されていると等しいとき s は s' とおおうという。次の条件がなりたつとき F_2 は F_1 とおおうという。

- (1) F_2 の初期状態 I_2 は、 F_1 の初期状態 I_1 とおおう。
- (2) F_1, F_2 の入力系列 $w_1 \dots w_l, y_1 \dots y_m$ に対し図5のようになる。

これら $L_1, \dots, L_m, \dots, L_1, \dots, L_m$ の全体とともにおおうような状態 L' が F_2 に存在する。

(3) 上のような L' の中で少なくとも一つの L' が存在して、図 5-(2) の I_2 を L' とし、同図 (1) の I_1 を上の $L_1, \dots, L_m, \dots, L_1, \dots, L_m$ の全部にわたって考えたときの図 5 における $L_1, \dots, L_m$ の全体とともにおおうような状態が F_2 に存在する。

(4) 上の (3) とより通し考えても、(3) はつねに成り立つ。

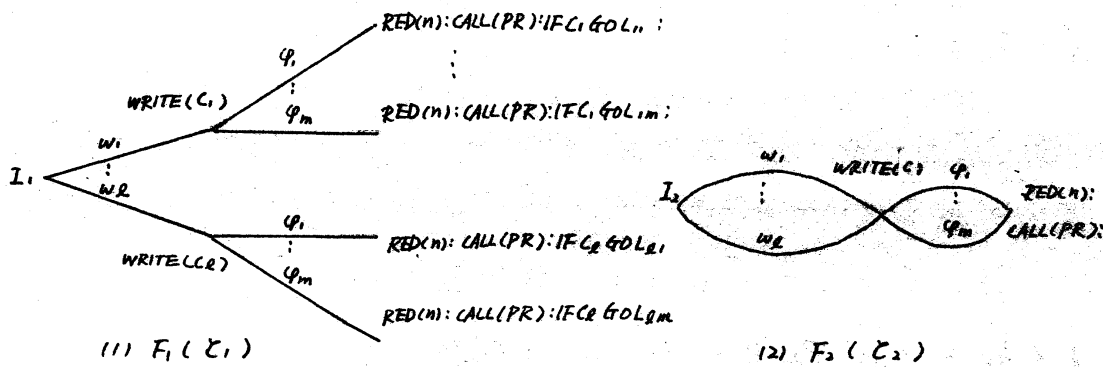


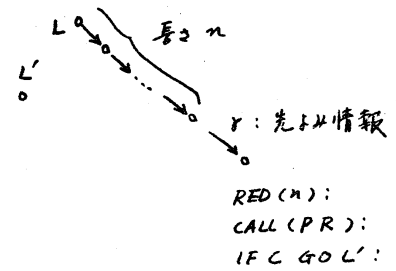
図 - 5

さて、次のことが証明される。

定理 3 P_2 が P_1 と等価であるための必要十分条件は、 F_2 が F_1 をおおうことである。

F_1 とおおう F_2 は、両立、許容の定義を適当に行なえば、 F_1 の状態の両立許容分解から得られる。すなわち、 P_1 の簡単化の問題は、順序機械における Paull - Unger の問題と類似の問題に帰着される。

とくに上の F_1, F_2 において L から L' へラベル $(n, (PR), r)$ を
 ヌツ枝をつけて得られる有限オート
 マトニと $\tilde{F}_1, \tilde{F}_2$ とする。



定理 4 $\tilde{F}_1, \tilde{F}_2$ が決定性の有限オートマトンならば、 P_2 が
 P_1 と等価であるための必要十分条件は、 $\tilde{F}_2$ の初期状態 I_2 が
 $\tilde{F}_1$ の初期状態 I_1 をおおうことである。

従って $\tilde{F}_2$ を求めることは、 $\tilde{F}_1$ の状態の通常の意味の両立許
 容分解を求めることにたなる。状態数最小の $\tilde{F}_2$ を求めることは、
 よく知られた Paull - Unger の問題になる。

5. あとがき

実際のコンパイラにおいては、パラメータだけ書きかえて、
 内部コードを発生しない動作もある。本稿では、簡単のため
 つぎの内部コード発生おびの何回かのパラメータの書きかえ
 を、一つのルーチン P で表現したが、パラメータ書きかえの
 ルーチンの入れ子構造や、セマンティックルーチン R へ代入し
 たとき、ある種の等価性を定義しておけば、本稿で述べたの
 と同じ意味でアナライザの等価性が決定でき、また同様な方

法でアナライザの簡単化ができる。

また、本稿では先ずは同じ部分を2度繰り返しては見ない (LH は左にもどらない) としたが、この制限をなくせば、プログラムがより短くなる可能性がある。いづれにせよ、計算時間と記憶容量の兼ね合いの問題である。

文 献

- (1) D.E. Knuth: "On the translation of languages from left to right", Inform. Control 8, 5, pp. 607-639, (Oct. 1965)
- (2) D. Pager: "A solution to an open problem by Knuth", Inform. Control 17, 5, pp. 462-473, (Dec. 1970)
- (3) F.L. DeRemer: "Simple LR(K) grammars", Comm. ACM 14, 7, pp. 453-460, (July 1971)
- (4) 林: "CFG-PL 変換について", 情報処理, 12, 3, pp. 145-153 (1971-03)
- (5) 雨宮: "LR(K) Parser について" 昭46 情報処理学会大会, 115.
- (6) M.C. Paull & S.H. Unger: "Minimizing the number of states in incompletely specified sequential switching functions", IRE Trans. Comput. EC-8, 3, pp. 356-366, (Sept 1959)
- (7) 谷口, 菊野, 嵩: "構文解析の簡単化について", 信学会オートマトン研資 (1972-01)